UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/037,592	01/02/2002	Bohuslav Rychlik	42390P13149	6341

8791 7590 08/13/2004

BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CA 90025-1030

EXAMINER	
HUISMAN, DAVID J	

ART UNIT	PAPER NUMBER
2183	

DATE MAILED: 08/13/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/037,592	RYCHLIK ET AL.	
	Examiner	Art Unit	
	David J. Huisman	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *02 January 2002*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-19* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-19* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *02 January 2002* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-19 have been examined.

Specification

2. The abstract of the disclosure is objected to because of the following minor informalities: The last two lines of the abstract seem to be grammatically incorrect. The examiner suggests rephrasing this portion of the abstract. Correction is required. See MPEP § 608.01(b).

3. The disclosure is objected to because of the following informalities: On page 1, line 14, replace "Itanium application" with --Itanium applications--. On page 16, line 10, replace "need" with --needs--.

Appropriate correction is required.

Claim Objections

4. Claim 1 is objected to because of the following informalities: The examiner recommends rephrasing lines 6-7 of the claim, as they do not seem to be grammatically correct. Appropriate correction is required.

5. Claim 3 is objected to because of the following informalities: Please insert --of-- before "register files" in line 4. Appropriate correction is required.

6. Claim 7 is objected to because of the following informalities: The examiner recommends rephrasing lines 1-2 of the claim (specifically the "said correlating busy condition" phrase), as they do not seem to be grammatically correct. Appropriate correction is required.

7. Claim 8 is objected to because of the following informalities: The examiner recommends rephrasing lines 10-11 of the claim, as they do not seem to be grammatically correct. Appropriate correction is required.

8. Claim 11 is objected to because of the following informalities: The examiner recommends rephrasing lines 9-10 of the claim, as they do not seem to be grammatically correct. Appropriate correction is required.

9. Claim 13 is objected to because of the following informalities: Please insert --of-- before "register files" in line 4. Appropriate correction is required.

Claim Rejections - 35 USC § 102

10. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

11. Claims 1-2, 6-7, and 11-12 are rejected under 35 U.S.C. 102(b) as being anticipated by Hennessy and Patterson, "Computer Architecture - A Quantitative Approach, 2nd Edition" 1996 (herein referred to as Hennessy).

12. Referring to claim 1, Hennessy has taught a method comprising:

a) allocating a plurality of registers. See page 247, Fig.4.4 (the functional unit status section). Note that each instruction is allocated a plurality of registers. For instance, the Div instruction is allocated registers F10, F0, and F6 for reading and writing purposes.

b) enabling execution of computer instructions concurrently by using the plurality of registers. See page 243 (the paragraph beginning with "The goal..."). Note that multiple instructions must be in their EX stage simultaneously (i.e., they execute concurrently).

c) tracking and reducing data dependencies in the computer instructions by correlating busy condition of a computer instruction to each register. See page 247, and note that the scoreboard uses flags Rj and Rk to track whether registers are busy or not. For instance, in Fig.4.4, in the "functional unit status" section, the multiply needs to read two sources (F2 and F4). However, Rj says that F2 is not ready. Therefore, the multiply instruction may not proceed. Also, see pages 232-234 and 252 and note that register renaming is an additional function that a system may perform. Doing so reduces dependencies. And, as is known in the art, dependencies may result in hazards which slow down the machine because stalling is required. Therefore, by reducing and/or eliminating dependencies, the system's performance will be improved.

13. Referring to claim 2, Hennessy has taught a method as described in claim 1. Hennessy has further taught that said enabling includes enabling execution of the computer instructions out of order. See pages 242-243 (starting with the scoreboard section).

14. Referring to claim 6, Hennessy has taught a method as described in claim 1. Hennessy has further taught that said allocating, enabling, and tracking includes substantially reducing computer instruction stalls due to data dependencies. Note from page 242 that scoreboarding is used in conjunction with out-of-order execution. The purpose of out-of-order execution, as discussed on pages 240-242 is to execute instructions as soon as they are ready, thereby reducing stalls due to data dependencies.

15. Referring to claim 7, Hennessy has taught a method as described in claim 1. Hennessy has further taught that said correlating busy condition of a computer instruction to each register includes providing each register with a corresponding scoreboard bit. Again, see page 247 and note that the scoreboard uses flags Rj and Rk to track whether registers are busy or not. As is known in the art, a flag is a value which may take on one of two values (a bit). And since this bit is part of the scoreboard, it is a scoreboard bit.

16. Referring to claim 11, Hennessy has taught a computer readable medium containing executable instructions (it is inherent that instructions exist on a computer readable medium) which, when executed in a processing system, causes the system to perform concurrent execution of computer instructions, comprising:

a) allocating a plurality of registers. See page 247, Fig.4.4 (the functional unit status section). Note that each instruction is allocated a plurality of registers. For instance, the Div instruction is allocated registers F10, F0, and F6 for reading and writing purposes.

b) enabling execution of computer instructions concurrently by using the plurality of registers. See page 243 (the paragraph beginning with "The goal..."). Note that multiple instructions must be in their EX stage simultaneously (i.e., they execute concurrently).

c) tracking and reducing data dependencies in the computer instructions by correlating busy condition of a computer instruction to each register. See page 247, and note that the scoreboard uses flags Rj and Rk to track whether registers are busy or not. For instance, in Fig.4.4, in the "functional unit status" section, the multiply needs to read two sources (F2 and F4). However, Rj says that F2 is not ready. Therefore, the multiply instruction may not proceed. Also, see page 252 and note that the scoreboard may take on additional functionality (making it a reservation

station) wherein register renaming is performed. Doing so eliminates WAW and WAR hazards because dependencies are eliminated.

17. Referring to claim 12, Hennessy has taught a medium as described in claim 11. Hennessy has further taught that said enabling includes enabling execution of the computer instructions out of order. See pages 242-243 (starting with the scoreboard section).

Claim Rejections - 35 USC § 103

18. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

19. Claims 8-10 are rejected under 35 U.S.C. 103(a) as being unpatentable over Nojiri, U.S. Patent No. 5,179,685, in view of Parady, U.S. Patent No. 5,933,627, and further in view of Hennessy, as applied above.

20. Referring to claim 8, Nojiri has taught a method, comprising:

a) allocating a plurality of register files if there is a task switch associated with a currently used register file, each register file including a plurality of registers. See column 3, lines 5-9 and line 15-21. Nojiri has not explicitly taught the task switch is performed if there are any pending writes in the currently used register file. However, Parady has taught such a concept. See the abstract and note that when a load instruction misses the cache, a task switch is performed. As is known a load instruction modifies a register in the register file, and therefore, if the load misses the cache, the load would be a pending write (the data cannot be written until the data is finally

retrieved). As is shown in Fig.3 and column 3, lines 44-49, of Parady, each task is associated with a different register file. Therefore, a switch in tasks would yield a new current register file. As discussed in the abstract, the switch is done because the cache miss is a long-latency event. That is, if a miss in the cache occurs, a main memory access must occur, which will take a much longer amount of time. Therefore, the switch is performed so that a different task may run while the main memory access is occurring. Consequently, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Nojiri such that a task switch is performed upon a load cache-miss. This will result in higher efficiency since the processor will not be idle while the main memory access occurs.

b) maintaining a select register that tracks the currently used register file. See column 3, lines 59-65, and Fig.2a, components 13 and 16.

c) Nojiri has not explicitly taught enabling execution of computer instructions concurrently by using the plurality of register files. However, Hennessy has taught that out-of-order execution requires that multiple instructions (using integer registers (Rx) and floating-point registers (Fx)) execute concurrently. See page 243 (the paragraph which begins with "The goal…") and page 247. Note that multiple instructions must be in their EX stage simultaneously (i.e., they execute concurrently). In turn, out-of-order execution allows later instructions to proceed with execution even if previous instructions are stalled. See page 241. Clearly, this would prevent the processor from always being idle when one instruction stalls, thereby resulting in higher throughput and more efficiency. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Nojiri to allow multiple instructions to execute concurrently.

d) Nojiri has not taught tracking and reducing data dependencies in the computer instructions by correlating busy condition of a computer instruction to each register. However, Hennessy has taught such a concept. See page 247, and note that the scoreboard uses flags Rj and Rk to track whether registers are busy or not. For instance, in Fig.4.4, in the "functional unit status" section, the multiply needs to read two sources (F2 and F4). However, Rj says that F2 is not ready. Therefore, the multiply instruction may not proceed. Also, see pages 232-234 and 252 and note that register renaming is an additional function that a system may perform. Doing so reduces dependencies. And, as is known in the art, dependencies may result in hazards which slow down the machine because stalling is required. Therefore, by reducing and/or eliminating dependencies, the system's performance will be improved. Consequently, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Nojiri to include tracking and reducing of data dependencies in the computer instructions by correlating busy condition of a computer instruction to each register.

21. Referring to claim 9, Nojiri in view of Parady and further in view of Hennessy has taught a method as described in claim 8. Nojiri has further taught maintaining a free file list that tracks currently available register files. See Fig.1, components 4, and column 3, lines 21-23.

22. Referring to claim 10, Nojiri in view of Parady and further in view of Hennessy has taught a method as described in claim 8. Although Nojiri has not explicitly taught that said allocating, enabling, and tracking includes substantially reducing computer instruction stalls due to data dependencies, Hennessy has taught such a concept. Note from page 242 that scoreboarding is used in conjunction with out-of-order execution. The purpose of out-of-order execution, as discussed on pages 240-242 is to execute instructions as soon as they are ready,

thereby reducing stalls due to data dependencies. Clearly, it would be desirable in any system, including Nojiri's to reduce stalling and therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to employ a system in Nojiri, such as a scoreboard and out-of-order execution, which substantially reduces computer instruction stalls due to data dependencies.

23. Claims 1-4, 6-7, and 11-16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Gschwind et al., U.S. Patent No. 6,513,109 (herein referred to as Gschwind) in view of Hennessy, as applied above.

24. Referring to claim 1, Gschwind has taught a method comprising:

a) allocating a plurality of registers. From the code snippets in column 3, it is clear that instructions use registers. Consequently, it is inherent that during the execution of such instructions, that registers are allocated to instructions for reading and writing.

b) enabling execution of computer instructions concurrently by using the plurality of registers. Note from Fig.4 that multiple functional units exist (340, 345, 350). And, from column 1, lines 28-34, it is disclosed that multiple functional units exist within superscalar machines in order to execute multiple instructions simultaneously.

c) Gschwind has not explicitly taught tracking and reducing data dependencies in the computer instructions by correlating busy condition of a computer instruction to each register. However, Hennessy has taught such a concept. See page 247, and note that the scoreboard uses flags Rj and Rk to track whether registers are busy or not. For instance, in Fig.4.4, in the "functional unit status" section, the multiply needs to read two sources (F2 and F4). However, Rj says that F2 is

not ready. Therefore, the multiply instruction may not proceed. Also, see pages 232-234 and 252 and note that register renaming is an additional function that a system may perform. Doing so reduces dependencies. And, as is known in the art, dependencies may result in hazards which slow down the machine because stalling is required. Therefore, by reducing and/or eliminating dependencies, the system's performance will be improved. Consequently, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Gschwind to include tracking and reducing of data dependencies in the computer instructions by correlating busy condition of a computer instruction to each register.

25. Referring to claim 2, Gschwind in view of Hennessy has taught a medium as described in claim 1. Gschwind has further taught that said enabling includes enabling execution of the computer instructions out of order. Note from the abstract that predicated instructions are executed out-of-order by predicting the predicate values using the future predicate register file (note that the future file is used to hold speculative or out-of-order results while the architected file holds the true results based on in-order execution - column 8, lines 4-7).

26. Referring to claim 3, Gschwind in view of Hennessy has taught a medium as described in claim 1. Gschwind has further taught allocating a plurality of register files, each register file including the plurality of registers, the plurality register files enabling keeping copies of register contents. See Fig.4 and note that multiple future register files (335 and 405) and multiple architected register files (360 and 410) are allocated to instructions for use within the system. In essence, a future file is used to hold speculative results while an architected file holds actual results. See column 8, lines 4-7. However, speculative results may be equal to the architected results, in which case, the contents of at least one register in the architected file would be equal

to the contents of a corresponding register in the future file. See Fig.6 and note that the speculative predicate is used to make a prediction (step 635). If the prediction happens to be true (step 660), then the speculative predicate is equal to the actual predicate.

27. Referring to claim 4, Gschwind in view of Hennessy has taught a system as described in claim 3. Gschwind has further taught maintaining a select register that tracks a currently used register file. See column 11, line 63, to column 12, line 14. Note that the architected register file is accessed first to see if the predicate is available. If not, an indication, which is stored in the file, will be made to select the predicate from the future file. Clearly the register file comprises registers and therefore, the indication would be stored in a register. This indicating register is the select register because when read, it results in selecting the future file, making the future file the current register file.

28. Referring to claim 6, Gschwind in view of Hennessy has taught a system as described in claim 1. Furthermore, Gschwind in view of Hennessy has taught that said allocating, enabling, and tracking includes substantially reducing computer instruction stalls due to data dependencies. Note from page 242 of Hennessy that scoreboarding (dependency tracking device) is used in conjunction with out-of-order execution, which is taught by Gschwind. The purpose of out-of-order execution, as discussed on pages 240-242 is to execute instructions as soon as they are ready, thereby reducing stalls due to data dependencies.

29. Referring to claim 7, Gschwind in view of Hennessy has taught a system as described in claim 1. Hennessy has further taught that said correlating busy condition of a computer instruction to each register includes providing each register with a corresponding scoreboard bit. Again, see page 247 and note that the scoreboard uses flags Rj and Rk to track whether registers

are busy or not. As is known in the art, a flag is a value which may take on one of two values (a bit). And since this bit is part of the scoreboard, it is a scoreboard bit.

30. Referring to claim 11, Gschwind has taught a computer readable medium containing executable instructions (Fig.4, component 310) which, when executed in a processing system, causes the system to perform concurrent execution of computer instructions, comprising:

a) allocating a plurality of registers. From the code snippets in column 3, it is clear that instructions use registers. Consequently, it is inherent that during the execution of such instructions, that registers are allocated to instructions for reading and writing.

b) enabling execution of computer instructions concurrently by using the plurality of registers. Note from Fig.4 that multiple functional units exist (340, 345, 350). And, from column 1, lines 28-34, it is disclosed that multiple functional units exist within superscalar machines in order to execute multiple instructions simultaneously.

c) Gschwind has not explicitly taught tracking and reducing data dependencies in the computer instructions by correlating busy condition of a computer instruction to each register. However, Hennessy has taught such a concept. See page 247, and note that the scoreboard uses flags Rj and Rk to track whether registers are busy or not. For instance, in Fig.4.4, in the "functional unit status" section, the multiply needs to read two sources (F2 and F4). However, Rj says that F2 is not ready. Therefore, the multiply instruction may not proceed. Also, see pages 232-234 and 252 and note that register renaming is an additional function that a system may perform. Doing so reduces dependencies. And, as is known in the art, dependencies may result in hazards which slow down the machine because stalling is required. Therefore, by reducing and/or eliminating dependencies, the system's performance will be improved. Consequently, it would have been

obvious to one of ordinary skill in the art at the time of the invention to modify Gschwind to include tracking and reducing of data dependencies in the computer instructions by correlating busy condition of a computer instruction to each register.

31. Referring to claim 12, Gschwind in view of Hennessy has taught a medium as described in claim 11. Gschwind has further taught that said enabling includes enabling execution of the computer instructions out of order. Note from the abstract that predicated instructions are executed out-of-order by predicting the predicate values using the future predicate register file (note that the future file is used to hold speculative or out-of-order results while the architected file holds the true results based on in-order execution - column 8, lines 4-7).

32. Referring to claim 13, Gschwind in view of Hennessy has taught a medium as described in claim 11. Gschwind has further taught allocating a plurality of register files, each register file including the plurality of registers, the plurality register files enabling keeping copies of register contents. See Fig.4 and note that multiple future register files (335 and 405) and multiple architected register files (360 and 410) are allocated to instructions for use within the system. In essence, a future file is used to hold speculative results while an architected file holds actual results. See column 8, lines 4-7. However, speculative results may be equal to the architected results, in which case, the contents of at least one register in the architected file would be equal to the contents of a corresponding register in the future file. See Fig.6 and note that the speculative predicate is used to make a prediction (step 635). If the prediction happens to be true (step 660), then the speculative predicate is equal to the actual predicate.

33. Referring to claim 14, Gschwind has taught a system, comprising:

a) a plurality of register files (see Fig.4, components 405 and 410), each register file comprising:

a1) a plurality of predicate registers (Fig.5, components 405 and 410) to enable execution of computer instructions concurrently and out of order. Note from the abstract that predicated instructions are executed out-of-order by predicting the predicate values using the future predicate register file (note that the future file is used to hold speculative or out-of-order results while the architected file holds the true results based on in-order execution - column 8, lines 4-7). Also, note from Fig.4 that multiple functional units exist (340, 345, 350). And, from column 1, lines 28-34, it is disclosed that multiple functional units exist within superscalar machines in order to execute multiple instructions simultaneously.

a2) Gschwind has not taught a plurality of scoreboard registers to track data dependencies among the computer instructions. However, Gschwind has taught the use of a reservation station. See column 10, lines 28-32. Hennessy, on the other hand, has not only taught that reservation stations are comparable to scoreboards (with additional functionality like register renaming), but he has taught that reservation stations include a field (register) which tracks dependencies. See pages 254-255. Note that fields Qj and Qk are used to specify the availability of the source operands. These are fields for tracking dependencies because, as seen in Fig.4.9 on page 256, the multiply instruction is receiving a source from the Load2 unit, and it cannot execute until the Load2 unit provides the source. Because this is the standard operation of a reservation station, it would have been obvious to one of ordinary skill in the art at the time of the invention include these features in the reservation station of Gschwind.

34. Referring to claim 15, Gschwind in view of Hennessy has taught a system as described in claim 14. Gschwind has further taught a select register to select a current register file from the plurality of register files. See column 11, line 63, to column 12, line 14. Note that the architected register file is accessed first to see if the predicate is available. If not, an indication will be made to select it from the future file. Clearly the register file comprises registers and therefore, the indication would be stored in a register. This indicating register is the select register because when read, it results in selecting the future file.

35. Referring to claim 16, Gschwind in view of Hennessy has taught a system as described in claim 15. Gschwind has further taught that the select register includes a pointer. Whatever the indication may be, it results in directing the system to access the future predicate file, and therefore, it is a pointer in the sense that it points the system to access the future file.

36. Claims 5 and 17-19 are rejected under 35 U.S.C. 103(a) as being unpatentable over Gschwind in view of Hennessy, as applied above, and further in view of Nojiri, as applied above.

37. Referring to claim 5, Gschwind in view of Hennessy has taught a system as described in claim 3. Gschwind has not taught maintaining a free file list that tracks currently available register files. However, Noriji has taught such a concept. See column 3, lines 21-23. Noriji's free list is used to track available register files which may be assigned to a particular task when a task switch is performed. This register file allocation-type system is useful in that data may be stored at the same time for a plurality of tasks without having to transfer data to and from memory, which is a time consuming process. See column 2, lines 57-60. Consequently, it

would have been obvious to one of ordinary skill in the art at the time of the invention to modify Gschwind to include a free list which assists in allocating resources to multiple tasks.

38. Referring to claim 17, Gschwind in view of Hennessy has taught a system as described in claim 15. Gschwind has not taught a free file list to maintain a list of available register files. However, Noriji has taught such a concept. See column 3, lines 21-23. Noriji's free list is used to track available register files which may be assigned to a particular task when a task switch is performed. This register file allocation-type system is useful in that data may be stored at the same time for a plurality of tasks without having to transfer data to and from memory, which is a time consuming process. See column 2, lines 57-60. Consequently, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Gschwind to include a free list which assists in allocating resources to multiple tasks.

39. Referring to claim 18, Gschwind in view of Hennessy and further in view of Noriji has taught a system as described in claim 17. Noriji has further taught that the free file list includes a pointer. See Fig.3, components 5.

40. Referring to claim 19, Gschwind in view of Hennessy and further in view of Noriji has taught a system as described in claim 17. Noriji has further taught that the free file list includes a stack. See Fig.3 and column 10, lines 46-52, and column 14, lines 41-55. More specifically, when a file is needed from the free list, the top one (the one pointer to by pointer 5f in Fig.3) is "popped" off and linked to the active register file set. However, when an active file is no longer needed, it is "pushed" onto the free list. Therefore, the list operates as a stack.

Conclusion

41. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. Applicant is reminded that in amending in response to a rejection of claims, the patentable novelty must be clearly shown in view of the state of the art disclosed by the references cited and the objections made. Applicant must also show how the amendments avoid such references and objections. See 37 CFR § 1.111(c).

Arora, U.S. Patent No. 6,442,678, has taught a method and apparatus for providing data to a processor pipeline. Arora has also disclosed multiple predicate register files; one being speculative and the other being actual.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to David J. Huisman whose telephone number is (703) 305-7811. The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (703) 305-9712. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

DJH
David J. Huisman
July 23, 2004

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100